ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ЗАТВЕРДЖУЮ
Декан факультету
інформаційних технологій
_____ Олег САВЕНКО
_____ 2023 р.

РОБОЧА ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ
Програмування

Галузь знань 12 – Інформаційні технології
Спеціальність 121 – Інженерія програмного забезпечення
Рівень вищої освіти – Перший бакалаврський
Освітньо-професійна програма – Інженерія програмного забезпечення
Обсяг дисципліни –8 кредитів ЄКТС, ***Шифр дисципліни*** – ОПП.02
Статус дисципліни: обов'язкова, ***Мова навчання*** Англійська, українська
Факультет – Інформаційних технологій
Кафедра – Інженерії програмного забезпечення

Форма здобуття освіти	Курс	Семестр	Загальне навантаження		Кількість годин						Курсовий проект	Курсова робота	Форма семестрового контролю	
					Аудиторні заняття				Індивідуальна робота студента	Самостійна робота, в т.ч. ІРС				
			Кредити ЄКТС	Години	Разом	Лекції	Лабораторні роботи	Практичні заняття					Залік	Іспит
Очна (денна)	1	1	8	240	119	34	68	17		121				+
Разом			**8**	**240**	**119**	**34**	**68**	**17**		**121**				**1**

Робоча програма складена на основі Стандарту вищої освіти, освітньо-професійної програми підготовки бакалаврів 2023 року та навчального плану.

Програма складена _____ доктор техн. наук, професор Валерій МАРТИНЮК
Схвалена на засіданні кафедри інженерії програмного забезпечення
Протокол № 7 від 29.05. 2023 р.
Зав.кафедри інженерії програмного забезпечення _____ Леонід БЕДРАТЮК
Робоча програма розглянута та схвалена вченою радою факультету інформаційних технологій
Голова вченої ради _____ Олег САВЕНКО

Хмельницький 2023

ПРОГРАМУВАННЯ

Тип дисципліни	Обов’язкова
Рівень вищої освіти	Перший (бакалаврський)
Мова викладання	Українська, Англійська
Семестр	Перший
Обсяг кредитів ЄКТС	8
Форма здобуття освіти	Очна (денна)

Результати навчання. Студент, який успішно завершив вивчення дисципліни, має: аналізувати, цілеспрямовано шукати і вибирати необхідні для вирішення професійних завдань інформаційно-довідникові ресурси і знання з урахуванням сучасних досягнень науки і техніки; знати і застосовувати на практиці фундаментальні концепції, парадигми і основні принципи функціонування мовних, інструментальних і обчислювальних засобів інженерії ПЗ; проводити передпроектне обстеження предметної області, системний аналіз об'єкта проектування; вибирати вихідні дані для проектування, керуючись формальними методами опису вимог та моделювання; застосовувати на практиці ефективні підходи щодо проектування ПЗ; мотивовано обирати мови програмування та технології розробки для розв’язання завдань створення і супроводження ПЗ; *аналізувати* вхідні дані завдання; *визначати* функціональні вимоги до розроблюваної програми; *аналізувати* і *порівнювати* методи розв'язання задачі та *обгрунтувати* обраний спосіб; *виконувати* алгоритмізацію задач та кодування; *інтерпретувати* результати роботи програми; *підбирати* тестові набори даних; *виявляти* та *усувати* програмні помилки; *оцінювати* ступінь відповідності розробленої програми визначеним вимогам.

Пререквізити – Вихідна

Кореквізити – Об’єктно-орієнтоване програмування, Бази даних

Зміст навчальної дисципліни. Основи програмування. Парадигми програмування. Алгоритми та розв’язання задач. Фундаментальні структури даних. Структурне програмування. Конструкції мов програмування. Рекурсія. Програмування динамічних структур даних. Алгоритми та структури даних. Виключення та їх обробка.

Запланована навчальна діяльність: лекції – 34 год., лабораторні заняття – 68 год., практичні заняття – 17 год., самостійна робота – 121 год., разом – 240 год.

Форми (методи) навчання: лекції (з використанням методів проблемного навчання і візуалізації); лабораторні та практичні заняття (з використанням методів інформаційних технологій та сучасних інтегрованих середовищ програмування, майстер-класів, практикумів), самостійна робота (домашні практичні завдання).

Форми і методи оцінювання результатів навчання: захист лабораторних робіт; презентація результатів виконання практичних завдань; тестування; контрольна робота; письмова підсумкова робота.

Форма семестрового контролю: іспит.

Навчальні ресурси:

1. Seacord R. Effective C: An Introduction to Professional C Programming. - San Francisco, California, No Starch Press, 2020. - 272 p.
2. Gustedt J. Modern C. - Shelter Island, Manning, 2020. – 408 p.
3. King K.N. C programming: a modern approach. - W. W. Norton & Company, 2020. – 832 p.
4. Harwani B.M. Practical C Programming: Solutions for modern C developers to create efficient and well-structured programs. – Birmingham, Packt Publishing, 2020. – 616 p.
5. Hubert H.W. Intermediate C programming for the PIC microcontroller: simplifying embedded programming. - Berkeley, CA, Apress L. P., 2020. – 318 p.4. Seacord R. Effective C: An Introduction to Professional C Programming. - San Francisco, California, No Starch Press, 2020. - 272 p.
6. Модульне середовище для навчання MOODLE. https://msn.khmnu.edu.ua.
7. Електронна бібліотека університету. Доступ до ресурсу: http://library.khmnu.edu.ua

Викладач: доктор технічних наук, професор Валерій МАРТИНЮК

3. ПОЯСНЮВАЛЬНА ЗАПИСКА

Дисципліна «Програмування» є однією із дисциплін професійної підготовки і займає провідне місце у підготовці фахівців освітнього рівня "бакалавр" за освітньо-професійною програмою "Інженерія програмного забезпечення".

Пререквізити – Вихідна
Кореквізити – Об'єктно-орієнтоване програмування, Бази даних

Мета дисципліни. Метою дисципліни «Програмування» є: навчити студентів сучасним методам розробки алгоритмів і програм та їх практичному використанню при проектуванні і розробці програмного забезпечення.

Предмет дисципліни. Теорія і практика застосування базових алгоритмічних структур, базових і похідних структур даних та засобів мови програмування високого рівня при розробці ПЗ.

Завдання дисципліни. Надати студентам знання і практичні навички з основ проектування та розробки ПЗ мовою високого рівня; сформувати базові компетентності (загальні і фахові), необхідні для подальшого вивчення циклу професійно-орієнтованих дисциплін.

Відповідно до ***Стандарту вищої освіти*** із та освітньої програми дисципліна сприяє забезпеченню:
компетентностей:
ЗК1 – здатність до абстрактного мислення, аналізу та синтезу;
ФК10. Здатність накопичувати, обробляти та систематизувати професійні знання щодо створення і супроводження програмного забезпечення та визнання важливості навчання протягом всього життя.
ФК13 – здатність обґрунтовано обирати та освоювати інструментарій з розробки та супроводження програмного забезпечення (ПЗ);
ФК14 – здатність до алгоритмічного та логічного мислення;
програмних результатів навчання:
ПРН1 – аналізувати, цілеспрямовано шукати і вибирати необхідні для вирішення професійних завдань інформаційно-довідникові ресурси і знання з урахуванням сучасних досягнень науки і техніки;
ПРН7 – знати і застосовувати на практиці фундаментальні концепції, парадигми і основні принципи функціонування мовних, інструментальних і обчислювальних засобів інженерії ПЗ;
ПРН10 – проводити передпроектне обстеження предметної області, системний аналіз об'єкта проектування;
ПРН11 – вибирати вихідні дані для проектування, керуючись формальними методами опису вимог та моделювання;
ПРН12 – застосовувати на практиці ефективні підходи щодо проектування ПЗ;
ПРН13 – Знати і застосовувати методи розробки алгоритмів, конструювання програмного забезпечення та структур даних і знань.
ПРН15 – мотивовано обирати мови програмування та технології розробки для розв'язання завдань створення і супроводження ПЗ.

Результати навчання. Студент, який успішно завершив вивчення дисципліни, має: аналізувати, цілеспрямовано шукати і вибирати необхідні для вирішення професійних завдань інформаційно-довідникові ресурси і знання з урахуванням сучасних досягнень науки і техніки;
знати і застосовувати на практиці фундаментальні концепції, парадигми і основні принципи функціонування мовних, інструментальних і обчислювальних засобів інженерії ПЗ; проводити

передпроектне обстеження предметної області, системний аналіз об'єкта проектування; вибирати вихідні дані для проектування, керуючись формальними методами опису вимог та моделювання; застосовувати на практиці ефективні підходи щодо проектування ПЗ; знати і застосовувати методи розробки алгоритмів, конструювання програмного забезпечення та структур даних і знань; мотивовано обирати мови програмування та технології розробки для розв'язання завдань створення і супроводження ПЗ; *аналізувати* вхідні дані завдання; *визначати* функціональні вимоги до розроблюваної програми; *аналізувати* і *порівнювати* методи розв'язання задачі та *обгрунтувати* обраний спосіб; *виконувати* алгоритмізацію задач та кодування; *інтерпретувати* результати роботи програми; *підбирати* тестові набори даних; *виявляти* та *усувати* програмні помилки; *оцінювати* ступінь відповідності розробленої програми визначеним вимогам.

Політика дисципліни Організація освітнього процесу з дисципліни відповідає вимогам положень про організаційне і навчально-методичне забезпечення освітнього процесу, освітній програмі та навчальному плану. Студент зобов'язаний відвідувати лекції, практичні заняття, лабораторні роботи, тощо, згідно з розкладом, не запізнюватися на заняття, виконувати усі завдання та контрольні точки відповідно до графіка. Пропущені практичні заняття і лабораторні роботи студент зобов'язаний опрацювати самостійно у повному обсязі і відзвітувати перед викладачем не пізніше, ніж за тиждень до чергової атестації. До практичних занять і лабораторних робіт студент має підготуватися за відповідною темою і проявляти активність. Набутті особою знання з дисципліни або її окремих розділів у неформальній освіті зараховуються відповідно до Положення про порядок перезарахування результатів навчання та визначення академічної різниці у ХНУ.

4. СТРУКТУРА ЗАЛІКОВИХ КРЕДИТІВ ДИСЦИПЛІНИ

Назва теми	Кількість годин, відведених на:			
	лекції	практичні роботи	лабораторні роботи	самостійну роботу
Перший семестр				
Тема 1. Інформаційні технології. Основи програмування. Парадигми програмування.	4	2	8	12
Тема 2. Основи програмування. Алгоритми та розв'язання задач.	4	2	8	12
Тема 3. Основи програмування. Фундаментальні структури даних.	6	3	12	24
Тема 4. Структурне програмування. Конструкції мов програмування.	14	7	28	46
Тема 5. Програмування динамічних структур даних. Алгоритми та структури даних.	6	3	12	25
Разом за 1-ий семестр:	**34**	**17**	**68**	**121**

5. ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

5.1 Зміст лекційного курсу

Номер лекції	Перелік змістових модулів, тем лекцій, їх анотації	Кількість годин
	Перший семестр	
	Тема 1. Інформаційні технології. Основи програмування. Парадигми програмування.	
1	***Лекція 1. Вступ. Інформаційні технології*** Вступ. Інформація та її подання. Двійкова арифметика. Літ. **[15, С.14-40]**	2
2	***Лекція 2. Вступ. Комп'ютерні системи*** Загальна структура комп'ютера. Комп'ютерні системи та їх складові. Прикладне програмне забезпечення. Системне програмне забезпечення. Парадигми програмування. Літ. **[15, С.14-40]**	2
	Тема 2. Основи програмування. Алгоритми та розв'язання задач.	
3	***Лекція 3. Алгоритми та їх властивості. Основні поняття*** Алгоритми і алгоритмізація. Поняття алгоритму. Алгоритми та їх властивості. Поняття про алгоритми та їх властивості. Властивості алгоритму. Метод покрокової деталізації. Форми подання алгоритмів. Літ. **[3, С.12-27]**	2
4	***Лекція 4. Алгоритми та їх властивості. Типи алгоритмів*** Запис алгоритму у вигляді блок-схем. Приклади. Базові алгоритмічні структури. Слідування. Розгалуження. Повторення. Приклади. Типи основних структур алгоритмів. Поняття програми. Літ. **[3, С.12-27]**	2
	Тема 3. Основи програмування. Фундаментальні структури даних.	
5	***Лекція 5. Введення в С*** Вступ до мови програмування С. Алфавіт та словник мови. Ідентифікатори. Базові типи даних. Модифікатори типів. Літ. **[1, С.7-18; 2, С. 6-12; 5-14]**	2
6	***Лекція 6. Структура програми мови С*** Константи. Префіксна та суфіксна форми. Вирази та операції. Десяткове, вісімкове та шістнадцяткове представлення. Ідентифікатори. Ключові слова. Коментарі. Стандарти мови С. Літ. **[1, С.7-18; 2, С. 13-20; 5-14]**	2
7	Лекція 7. ***Структура програми мови С. Стандартні функції*** Структура програми мови Паскаль. Прототипи функцій. Директиви для включення вмісту файлів. Основні засоби введення-виведення. Специфікації виведення функцій. Введення	2

	даних з використанням стандартних функцій. Літ. **[1, С.19-30; 2, С. 27-48; 5-14]**	
	Тема 4. Структурне програмування. Конструкції мов програмування.	
8	***Лекція 8. Загальні відомості про оператори*** Загальні відомості про оператори. Складений оператор. Порожній оператор. Вирази. Знак операції. Типи арифметичних операцій. Пріоритет операцій. Літ. **[1, С.31-111; 2, С. 17-26; 5-14]**	2
9	***Лекція 9. Загальні відомості про оператори*** Оператор розгалуження, його форми. Логічні операції. Складені оператори присвоювання. Операція вибору за умовою. Літ. **[1, С.85-111; 2, С. 17-27; 5-14]**	2
10	***Лекція 10. Загальні відомості про оператори*** Використання коми. Оператори циклу. Оператор з параметром та його особливості. Оператори циклу-поки та циклу-до, їх відмінності. Літ. **[1, С.85-111; 2, С. 17-27; 4-14]**	2
11	***Лекція 11. Функції в мові С*** Оголошення і визначення функцій. Аргументи, параметри, приклади. Передача параметрів за значенням та за посиланням. Повернення з функції та повернення значень. Арифметичні функції. Рекурсія. Рекурсивні виклики. Пряма та непряма рекурсії. Основні принципи структурного програмування. Різні типи функцій. Літ. **[1, С.183-234; 2, С. 27-47; 4-14]**	2
12	***Лекція 12. Масиви*** Опис масивів. Масиви елементів. Оголошення масивів. Доступ до компонентів масиву, одно- і n-мірні масиви. Приклади. Методи сортування та пошуку. Обчислювальна складність алгоритмів. Літ. **[1, С.125-140; 2, С. 29; 3, С.44-59, 105-117; 4-14]**	2
13	***Лекція 13. Рядки*** Рядки. Поняття та оголошення. Індекси елементів рядка. Обєднання рядків. Порівняння рядків. Введення – виведення рядків. Інші функції для обробки рядків. Приклади. Літ. **[1, С.141-163; 2, С. 52-53; 4-14]**	2
14	***Лекція 14. Файли*** Файли. Визначення. Типи файлів. Загальні правила для всіх типів файлів. Текстові файли. Алгоритм створення текстового файлу, читання, запис, дозапис в текстовий файл. Особливості застосування стандартних функцій і процедур при роботі з файлами. Літ. **[1, С.311-362; 2, С. 54-60; 4-14]**	2

	Тема 5. Програмування динамічних структур даних. Алгоритми та структури даних.	
15	***Лекція 15. Вказівники.*** Поняття вказівників. Створення вказівників. Оголошення вказівників. Вказівники і типи змінних. Вказівники і масиви. Передача масивів в функції. Динамічні структури даних. Літ. **[1, С.112-124; 2, С. 34-35; 4-14]**	2
16	***Лекція 16. Структури, об'єднання і нестандартні типи даних.*** Найпростіші структури. Складні структури. Масиви структур. Літ. **[1, С.112-124; 2, С. 30-31; 4-14]**	2
17	***Лекція 17. Додаткові відомості про вказівники*** Вказівники на вказівники. Відмінність динамічних даних від статичних. Стек, черга, двійкове дерево. Літ. **[1, С.246-295; 2, С. 35; 4-14]**	2
	Разом за 1-ий семестр:	34

5.2 Зміст лабораторних занять

№ з/п	Тема лабораторного заняття	Кількість годин
	Перший семестр	
1	Лінійні алгоритми. Реалізація мовою С.	8
2	Розгалужені алгоритми. Реалізація мовою С.	8
3	Програмування циклічних алгоритмів. Реалізація мовою С.	8
4	Масиви. Робота з одномірними масивами. Реалізація мовою С.	8
5	Двомірні масиви. Реалізація мовою С.	8
6	Функції. Реалізація мовою С.	8
7	Рядки. Реалізація мовою С.	8
8	Потоки та файли. Робота з текстовими файлами. Їх реалізація мовою С.	8
9	Підсумкове заняття.	4
	Разом за 1-ий семестр:	68

5.3 Зміст практичних занять

№ з/п	Тема практичного заняття	Кіль-кість годин
	Перший семестр	
1	***Практичні роботи №1-2.*** Лінійні алгоритми.	2
2	***Практичні роботи №3-4.*** Розгалужені алгоритми.	2
3	***Практичні роботи №5-6.*** Програмування циклічних алгоритмів.	2
4	***Практичні роботи №7-8.*** Масиви. Робота з одновимірними масивами.	2
5	***Практичні роботи №9-10.*** Двовимірні масиви.	2
6	***Практичні роботи №11-12.*** Функції.	2
7	***Практичні роботи №13-14.*** Рядки.	2
8	***Практичні роботи №15-16.*** Потоки та файли. Робота з текстовими файлами. Структури.	2
9	***Практична робота №17.*** Підсумкове заняття.	1
	Разом за 1-ий семестр:	17

5.4 Зміст самостійної (індивідуальної) роботи

Обсяг самостійної роботи з дисципліни “Програмування” становить 164 годин. Він включає опрацювання лекційного матеріалу, підготовку до виконання лабораторних робіт і їх захисту, підготовку до виконання практичних робіт, підготовку до поточного контролю.

Номер тижня	Вид самостійної роботи	К-ть годин
	Перший семестр	
1	Опрацювання лекційного матеріалу. Підготовка до лабораторної роботи №1 та практичної роботи №1. Самостійна робота над розробкою програми до лабораторної роботи №1.	21
2	Опрацювання лекційного матеріалу. Підготовка до лабораторної роботи №2 та практичної роботи №2. Самостійна робота над розробкою програми до лабораторної роботи №2.	21
3	Опрацювання лекційного матеріалу. Підготовка до лабораторної роботи №3 та практичної роботи №3. Самостійна робота над розробкою програми до лабораторної роботи №3.	21
4	Опрацювання лекційного матеріалу. Підготовка до лабораторної роботи №4 та практичної роботи №4. Самостійна робота над розробкою програми до лабораторної роботи №4.	21
5	Опрацювання лекційного матеріалу. Підготовка до лабораторної роботи №5 та практичної роботи №5. Самостійна робота над розробкою програми до лабораторної роботи №5.	21
6	Опрацювання лекційного матеріалу. Підготовка до лабораторної роботи №6 та практичної роботи №6. Самостійна робота над розробкою програми до лабораторної роботи №6.	21
7	Опрацювання лекційного матеріалу. Підготовка до лабораторної роботи №7 та практичної роботи №7. Самостійна робота над розробкою програми до лабораторної роботи №7.	21
8	Опрацювання лекційного матеріалу. Підготовка до лабораторної роботи №8 та практичної роботи №8. Самостійна робота над розробкою програми до лабораторної роботи №8.	14
	Разом за 1-ий семестр:	161

6. МЕТОДИ НАВЧАННЯ

Процес навчання з дисципліни ґрунтується на використанні традиційних та сучасних методів. Зокрема, використовувані методи: лекції (з використанням методів проблемного навчання і візуалізації); лабораторні та практичні заняття (з використанням методів інформаційних технологій та сучасних інтегрованих середовищ програмування, майстер-класів, практикумів), самостійна робота (домашні практичні завдання) , і які мають за мету – засвоєння студентами сучасних методів розробки алгоритмів і програм та їх практичному використанню при проектуванні і розробці програмного забезпечення.

7. ОЦІНЮВАННЯ РЕЗУЛЬТАТІВ НАВЧАННЯ СТУДЕНТІВ

Поточний контроль здійснюється під час лекційних, практичних та лабораторних занять, а також у дні проведення контрольних заходів, встановлених робочим планом дисципліни. Семестрові контролі проводяться у формі іспиту. При цьому при виведенні остаточної оцінки враховуються результати поточного контролю.

При викладанні дисципліни використовуються такі види навчальних занять, як лекції, лабораторні роботи, практичні роботи, індивідуальне консультування і керівництво самостійною роботою студента.

Кожний, обов'язковий для оцінювання, вид роботи з дисципліни оцінюється за *чотирибальною* шкалою. Семестрова підсумкова оцінка визначається як середньозважена з усіх видів навчальної роботи, виконаних і зданих *позитивно* з врахуванням коефіцієнта вагомості. Вагові коефіцієнти змінюються залежно від структури дисципліни і важливості окремих її видів робіт. Студент, який набрав позитивний середньозважений бал за поточну роботу і не здав підсумковий контрольний захід (іспит), вважається невстигаючим.

При оцінюванні знань студентів використовуються різні засоби контролю, зокрема: усне опитування перед допуском до виконання лабораторної роботи – здійснюється на її початку; засвоєння теоретичного матеріалу з тем перевіряється тестовим контролем; якість виконання, набуття теоретичних знань і практичних навичок перевіряється шляхом захисту кожної лабораторної роботи згідно з робочою програмою дисципліни і робочим навчальним планом.

Оцінка, яка виставляється за *лабораторне заняття*, складається з таких елементів: усне опитування студентів перед допуском до виконання лабораторної роботи; знання теоретичного матеріалу з теми; якість оформлення протоколу і графічної частини; вміння студента обґрунтувати прийняті конструктивні рішення; своєчасний захист лабораторної роботи. Для виконання програми дисципліни студент повинен отримати 8 оцінок за лабораторні роботи.

Термін захисту лабораторної роботи вважається своєчасним, якщо студент захистив її на наступному після виконання роботи занятті.

Пропущене лабораторне заняття студент повинен відпрацювати в лабораторіях кафедри у встановлений викладачем термін, але не пізніше, ніж за два тижні до кінця теоретичних занять у семестрі.

При *оцінюванні знань* студентів викладач керується такими критеріями.

Оцінку „відмінно", за шкалою ECTS – A (див. шкалу оцінок), отримує студент за глибоке і повне опанування змісту навчального матеріалу, в якому він легко орієнтується, понятійного апарату, за уміння зв'язувати теорію з практикою, вирішувати практичні завдання, висловлювати і обґрунтовувати свої судження. Відмінна оцінка передбачає грамотний, логічний виклад відповіді (як в усній, так і в письмовій формі), якісне зовнішнє оформлення. Студент повинен набути практичних навичок із складання різних алгоритмів та розробки програм за цими алгоритмами. Оцінка "відмінно" виставляється студенту, який глибоко засвоїв оператори, функції та процедури мови С та вміє їх раціонально застосувати, знає методики та вміє ними користуватися при складанні алгоритмів та програм. Студент не повинен вагатися при видозміні запитання, повинен робити детальні та узагальнюючі висновки.

Оцінку „добре", за шкалою ECTS – B, отримує студент за повне засвоєння навчального матеріалу, володіння понятійним апаратом, орієнтування в вивченому матеріалі, свідоме використання знань для вирішення практичних завдань, грамотний виклад відповіді, але у змісті і формі відповіді мали місце окремі неточності (похибки), нечіткі формулювання закономірностей тощо. Відповідь студента повинна будуватись на основі самостійного мислення.

Оцінку „добре", за шкалою ECTS – C, отримує студент за правильну відповідь з однією суттєвою помилкою.

Оцінки "задовільно", за шкалою ECTS – D, заслуговує студент, який виявив знання основного навчально-програмного матеріалу в обсязі, необхідному для подальшого навчання та практичної діяльності за професією, що справляється з виконанням практичних завдань, передбачених програмою. Як правило, відповідь студента будується на рівні репродуктивного мислення, студент слабо знає структуру курсу, допускає помилки у відповіді, засвоїв і набув практичних навичок у складанні програм, але допустив неточності. Вагається при відповіді на видозмінене запитання, разом з тим студент володіє знаннями, що дозволяють йому під керівництвом викладача усунути неточності у відповіді.

Оцінки "задовільно", за шкалою ECTS – E, заслуговує студент за неповне опанування програмного матеріалу, але ним отримані знання і набуті практичні навички із розробки програм мовою С.

Оцінка „незадовільно", за шкалою ECTS – FX, виставляється, коли студент має розрізнені, безсистемні знання, не вміє виділяти головне і другорядне, допускається помилок у визначенні

понять, перекручує їх зміст, хаотично і невпевнено викладає матеріал, не може використовувати знання при вирішенні практичних завдань. Як правило, оцінка "незадовільно" виставляється студенту, який не може продовжити навчання без додаткових знань з курсу.

Оцінка „незадовільно", за шкалою ECTS – F, виставляється студенту за повне незнання і нерозуміння навчального матеріалу або відмову від відповіді і передбачає повторне навчання студента з дисципліни.

Кожний вид роботи оцінюється за чотирибальною шкалою. Семестрова підсумкова оцінка визначається як середньозважена з усіх видів робіт.

Структурування дисципліни за видами робіт і оцінювання результатів навчання студентів у семестрі за ваговими коефіцієнтами

Аудиторна робота	**Самостійна робота**		**Семестровий контроль**
Лабораторні роботи	Тестовий контроль	Контрольна робота	іспит
№ 1 – 12	ТК1 (Т1-3) / ТК2 (Т4-6)	1	1
ВК: 0,3	**0,2**	**0,1**	**0,4**

Оцінювання тестових завдань. Тематичний тест для кожного студента складається з двадцяти п'яти тестових завдань, кожне з яких оцінюється одним балом. Максимальна сума балів, яку може набрати студент, складає 25.

Оцінювання здійснюється за чотирибальною шкалою.

Відповідність набраних балів за тестове завдання оцінці, що виставляється студенту, представлена у нижченаведеній таблиці.

Сума балів за тестове завдання	1–11	12–14	15–22	23-25
Оцінка	2	3	4	5

На тестування відводиться 30 хвилин. Тестування проводиться з використанням модульного середовища для навчання MOODLE. Правильні відповіді студент реєструє в он-лайн режимі в модульному середовищі MOODLE. Через 30 хвилин студенти завершують тестування та надсилають свої відповіді на сервер. Викладач оголошує результати тестування згідно журналу оцінок модульного середовища MOODLE.

Якщо студент отримав негативну оцінку, то він має перездати її в установленому порядку, але обов'язково до терміну наступного контролю.

Підсумкова семестрова оцінка за національною шкалою і шкалою ЄКТС встановлюється в автоматизованому режимі після внесення усіх оцінок до електронного журналу. Співвідношення вітчизняної шкали оцінювання і шкали оцінювання ЄКТС наведені у наступній таблиці.

Для переходу від вітчизняної оцінки до оцінки за шкалою ECTS необхідно знайти середньоарифметичну оцінку за вітчизняною шкалою, помножити її на відповідний ваговий коефіцієнт і, додавши всі складові, отримаємо суму балів, які визначать конкретну оцінку ECTS.

Співвідношення вітчизняної шкали оцінювання і шкали оцінювання ЄКТС

Оцінка ECTS	Бали	Вітчизняна оцінка	
A	4,75-5,00	5	ВІДМІННО – глибоке і повне опанування навчального матеріалу і виявлення відповідних умінь та навиків
B	4,25-4,74	4	ДОБРЕ – повне знання навчального матеріалу з кількома незначними помилками
C	3,75-4,24	4	ДОБРЕ – в загальному правильна відповідь з однією суттєвою помилкою
D	3,25-3,74	3	ЗАДОВІЛЬНО – неповне опанування програмного матеріалу, але достатнє для практичної діяльності за професією

E	3,00-3,24	3	ЗАДОВІЛЬНО – неповне опанування програмного матеріалу, що задовольняє мінімальні критерії оцінювання
FX	2,00 -2,99	2	НЕЗАДОВІЛЬНО – безсистемність одержаних знань і неможливість продовжити навчання без додаткових знань з дисципліни
F	0,00-1, 99	2	НЕЗАДОВІЛЬНО – необхідна серйозна подальша робота і повторне вивчення дисципліни

Залік виставляється при отриманні студентом з дисципліни від 3,00 до 5,00 балів. При цьому за вітчизняною шкалою ставиться «зараховано», а за шкалою ECTS – оцінка, що відповідає набраній студентом кількості балів.

8. ПИТАННЯ ДЛЯ САМОКОНТРОЛЮ СТУДЕНТІВ

1. Інформаційні технології та системи
2. Історія розвитку комп'ютерної техніки
3. Історія розвитку операційних систем, їх основні поширені реалізації
4. Інформація та її подання
5. Двійкова арифметика
6. Загальна структура комп'ютера
7. Комп'ютерні системи та їх складові
8. Поняття про алгоритми та їх властивості
9. Метод покрокової деталізації
10. Запис алгоритму у вигляді блок-схем
11. Форми подання алгоритмів
12. Типи основних структур алгоритмів
13. Алфавіт та словник мови С
14. Базові типи даних
15. Константи
16. Ідентифікатори
17. Ключові слова
18. Коментарі
19. Стандарти мови С
20. Структура програми мови С
21. Директива #include
22. Основні засоби введення-виведення
23. Загальні відомості про оператори
24. Вирази
25. Оператор розгалуження
26. Логічні операції
27. Складені оператори присвоювання
28. Операція вибору за умовою
29. Використання коми
30. Оператори циклу
31. Цикл з післяумовою, приклад програми
32. Цикл FOR
33. Цикл з передумовою, приклад програми
34. Оголошення і визначення функцій
35. Рекурсивний виклик
36. Масиви елементів
37. Багатомірні масиви
38. Метод попарної перестановки елементів
39. Метод найменших елементів
40. Метод бінарного пошуку
41. Поняття вказівників
42. Створення вказівників
43. Оголошення вказівників
44. Вказівники і типи змінних

45. Вказівники і масиви
46. Передача масивів в функції
47. Поняття про рядки
48. Способи виділення пам’яті
49. Функції введення і виведення символів та рядків
50. Операції інкременту та декременту
51. Оператор continue
52. Оператор break
53. Повернення значень функції за допомогою оператора return
54. Прототипи
55. Додаткові операції присвоювання
56. Файли об’єктного коду, виконувані файли і бібліотеки
57. Оператор розгалудження IF
58. Складені літерали.
59. Взаємодія з файлами.
60. Стандартні файли.
61. Функція fopen().
62. Програма зтискування файлів.
63. Введення – виведення файлів.
64. Довільний доступ до файлів
65. Найпростіші структури даних
66. Складні структури.
67. Масиви структур.
68. Ініціалізація структур.

9. МЕТОДИЧНЕ ЗАБЕЗПЕЧЕННЯ

Освітній процес з дисципліни « Програмування» повністю і в достатній кількості забезпечений необхідною навчально-методичною літературою.

10. РЕКОМЕНДОВАНА ЛІТЕРАТУРА

Основна література

1. Seacord R. Effective C: An Introduction to Professional C Programming. - San Francisco, California, No Starch Press, 2020. - 272 p.
2. Gustedt J. Modern C. - Shelter Island, Manning, 2020. – 408 p.
3. King K.N. C programming: a modern approach. - W. W. Norton & Company, 2020. – 832 p.
4. Harwani B.M. Practical C Programming: Solutions for modern C developers to create efficient and well-structured programs. – Birmingham, Packt Publishing, 2020. – 616 p.
5. Hubert H.W. Intermediate C programming for the PIC microcontroller: simplifying embedded programming. - Berkeley, CA, Apress L. P., 2020. – 318 p.
6. Winkle L.Van. Hands-On Network Programming with C: Learn socket programming in C and write secure and optimized network code. – Birmingham, Packt Publishing, 2019. – 478 p.
7. Harwani B.M. C programming cookbook: over 40 recipes exploring data structures, pointers, interprocess communication, and database in C. - Birmingham, UK, Packt Publishing, 2019. – 344 p.
8. Joyce P. Numerical C: applied computational programming with case studies. - Berkeley, CA, Apress, 2019. – 312 p.
9. McGrath M. C Programming in easy steps: Updated for the GNU Compiler version 6.3.0 and Windows 10 5th Edition. - In Easy Steps Limited, 2018. – 192 p.
10. Kane W, C programming for business. Book one, Introduction to problem solving in C language. - Choice Publishing, 2017. – 284 p.

Додаткова література

11. Harwani B.M. C programming cookbook: over 40 recipes exploring data structures, pointers, interprocess communication, and database in C. - Birmingham, UK, Packt Publishing, 2019. – 344 p.
12. Joyce P. Numerical C: applied computational programming with case studies. - Berkeley, CA, Apress, 2019. – 312 p.
13. McGrath M. C Programming in easy steps: Updated for the GNU Compiler version 6.3.0 and Windows 10 5th Edition. - In Easy Steps Limited, 2018. – 192 p.
14. Kane W, C programming for business. Book one, Introduction to problem solving in C language. - Choice Publishing, 2017. – 284 p.
15. Стандарт C11 http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1570.pdf

11. ІНФОРМАЦІЙНІ РЕСУРСИ

1. Модульне середовище для навчання. Доступ до ресурсу: https://msn.khmnu.edu.ua.
2. Електронна бібліотека університету. Доступ до ресурсу: http://library.khmnu.edu.ua
3. Репозитарій ХНУ:https://elar.khmnu.edu.ua/home

KHMELNYTSKYI NATIONAL UNIVERSITY

APPROVED
Dean of the Faculty Savenko O.S.
"1" 09 2023.

COURSE PROGRAM

Programming

Field of study 12 – Information technology
Major 121 – Software engineering
Educational program Software engineering
Course status: compulsory, training discipline
Faculty – Information technologies
Department – Software engineering

Study mode	Year	Semester	Total load		Number of hours						Course project	Course paper	Type of semester control	
					Classwork hours				Student's individual work	Student's independent work including individual work				
			ECTS credits	Hours	Total	Lectures	Laboratory works	Practical classes					Pass/ fail test	Examination
C	1	1	8	240	119	34	68	17		121				+
Total			**8**	**240**	**119**	**34**	**68**	**17**		**121**				**1**

The course program is based on the educational and professional program for bachelors majoring in "Software engineering"

Program's author _____ V.V. Martynyuk
Signature Initials, surname

Approved at the staff meeting of the department of Software engineering

Record № 7 of 29.05. 2023

Head of the Department of Software engineering _____ L. P. Bedratyuk
Signature Initials, surname

The course program is approved by the Academic Board of the Faculty of Information technologies

Head of the Academic Board _____ O.S. Savenko

Khmelnytskyi 2023

PROGRAMMING

Type of Discipline	Compulsory
Level of Higher Education	First (Bachelor's)
Language of Instruction	Ukrainian, English
Semester	1
ECTS Credits	8
Course study mode	Full-time (Daytime)

Learning outcomes. According to the Standard of higher education and the educational program, the discipline must provide:
Competencies: the ability for abstract thinking, analysis, and synthesis; the ability to make informed choices and master the tools for developing and maintaining software; the ability for algorithmic and logical thinking;
Program learning outcomes: to analyze, purposefully search for and select the necessary informational and reference resources and knowledge for solving professional tasks, taking into account modern achievements in science and technology; to know and apply in practice the fundamental concepts, paradigms, and main principles of the functioning of languages, tools, and computational means in software engineering; to conduct a pre-design survey of the subject area, a systematic analysis of the design object; to select input data for design, guided by formal methods of requirements description and modeling; to apply effective approaches to software design in practice; to motivatedly choose programming languages and development technologies for solving tasks of creating and maintaining SE.
Course content. Basics of programming. Programming paradigms. Algorithms and problem solving. Fundamental data structures. Structural programming. Constructions of programming languages. Recursion. Programming dynamic data structures. Algorithms and data structures. Exclusions and their processing.

Planned academic activity: lectures – 34 hrs, practical classes – 17 hrs, laboratory works – 68 hrs, independent work – 121 hrs, total – 240hrs.

Teaching forms (methods): verbal, visual, practical, problem-based, interactive methods, use of information technology, partial search.

Assessment forms and methods: oral examination, written tests, defense of laboratory work, testing.

Form of semester control: exam.

Educational resources:
1. Seacord R. Effective C: An Introduction to Professional C Programming. - San Francisco, California, No Starch Press, 2020. - 272 p.
2. Gustedt J. Modern C. - Shelter Island, Manning, 2020. – 408 p.
3. King K.N. C programming: a modern approach. - W. W. Norton & Company, 2020. – 832 p.
4. Harwani B.M. Practical C Programming: Solutions for modern C developers to create efficient and well-structured programs. – Birmingham, Packt Publishing, 2020. – 616 p.
5. Hubert H.W. Intermediate C programming for the PIC microcontroller: simplifying embedded programming. - Berkeley, CA, Apress L. P., 2020. – 318 p.
6. MOODLE Learning Platform. Access to the resource: https://msn.khmnu.edu.ua

Lecturer: Doctor of Technical Sciences, Professor V.V. Martynyuk

3. EXPLANATORY NOTE

The aim and tasks of the course. The discipline is one of the fundamental and professional disciplines and therefore occupies a leading position in the preparation of bachelors.

The purpose of the discipline is: 1) the formation of competencies necessary for abstract thinking, analysis and synthesis in the implementation of algorithmization and programming; 2) development of students' professional style of thinking on algorithmization; 3) providing knowledge of the theory of algorithms and constructions of programming languages (on the example of C), necessary for further study of special disciplines and for practical engineering activities; 4) developing students' ability to use the acquired knowledge in program development.

The subject of the discipline. Algorithmization and programming in C / C ++.

Tasks of discipline. To provide students with knowledge about algorithmization and its use for solving practical problems, to teach programming in C / C ++.

After studying the discipline "Programming" the student must achieve the following learning outcomes:

Integral competence: Ability to solve complex specialized tasks or practical problems of software engineering, characterized by complexity and uncertainty of conditions, using theories and methods of information technology.

GC1. Ability to abstract thinking, analysis and synthesis.

PC13. Ability to reasonably choose and master software development and maintenance tools.

PC14. Ability to algorithmic and logical thinking.

PLO1. To analyse, purposefully search for, and select the necessary information, reference resources, and knowledge for solving professional tasks, considering modern scientific and technical achievements.

PLO7. To know and apply in practice the fundamental concepts, paradigms and basic principles of the functioning of linguistic, instrumental and computing tools of software engineering.

PLO10. Conduct a pre-project survey of the subject area, system analysis of the design object.

PLO11. Select input data for design, guided by formal requirements description and modeling methods.

PLO12. Apply effective software design approaches in practice.

PLO15. To make informed decisions when choosing programming languages and development technologies to address the tasks of creating and maintaining software.

Learning Outcomes. After studying the discipline, the student should be able to: *skillfully use* the conceptual framework; *analyze* the input data of the task; *determine* the functional requirements for the developed program; *analyze and compare* the methods of solving the problem and *justify* the chosen approach; *perform* task algorithmization and coding, taking into account the features of basic algorithmic structures; *reduce* the solution of the problem to the solution of subtasks; *execute* program construction; *demonstrate* a culture of thinking in developing algorithms for subtasks and the task as a whole; *generalize* data when writing program code; *interpret* the results of the program's work; *select* test data sets; *identify and correct* software errors; *evaluate* the degree of correspondence of the developed program to the defined requirements.

Discipline Policy. The organization of the educational process for the discipline complies with the requirements of the provisions on organizational and instructional-methodological support of the educational process, the educational program, and the curriculum. Students are required to attend lectures, practical classes, laboratory work, etc., according to the schedule, not to be late for classes, and to complete all tasks and checkpoints according to the schedule. Missed practical classes and laboratory work must be independently completed by the student in full and reported to the instructor no later than one week before the next assessment. For practical classes and laboratory work, students must prepare on the relevant topic and demonstrate active participation. Knowledge acquired by an individual in the discipline or its specific sections through informal education is credited according to the Regulation on the procedure for transferring learning outcomes and determining academic differences at KhNU.

4. COURSE CREDIT STRUCTURE

Topic title	Number of hours for:			
	Full-time			
	Lectures	Practical classes	Laboratory works	Independent work
First semester				
Topic 1. Information Technology. Basics of programming. Programming paradigms.	4	2	8	12
Topic 2. Basics of programming. Algorithms and problem solving.	4	2	8	12
Topic 3 Basics of programming. Fundamental data structures.	6	3	12	24
Topic 4. Structural programming. Constructions of programming languages.	14	7	28	46
Topic 5. Programming dynamic data structures. Algorithms and data structures.	6	3	12	25
Total for the 1th semester:	**34**	**17**	**68**	**121**

5. COURSE PROGRAM

5.1. Content of lectures

Lecture number	Lecture topics and abstracts ***First semester***	Number of hours
	Topic 1. Information Technology. Basics of programming. Programming paradigms.	
1	**Lecture 1. Introduction. Information Technology.** Introduction. Information and its presentation. Binary arithmetic. Lit.: [**15, C.14-40**]	2
2	**Lecture 2. Introduction. Computer systems.** The general structure of the computer. Computer systems and their components. Application software. System software. Programming paradigms. Lit.: [**15, C.14-40**]	2
	Topic 2. Information Technology. Basics of programming. Programming paradigms.	
3	**Lecture 3. Algorithms and their properties. Basic concepts.** Algorithms and algorithmization. The concept of algorithm. Algorithms and their properties. The concept of algorithms and their properties. Algorithm properties. Step-by-step detailing method. Forms of algorithms. Lit.: [**3, C.12-27**]	2
4	**Lecture 4. Algorithms and their properties. Types of algorithms.** Recording of the algorithm in the form of block diagrams. Examples. Basic algorithmic structures. Following. Branching. Repetition. Examples. Types of basic structures of algorithms. The concept of the program. Lit.: [**3, C.12-27**]	2
	Topic 3. Basics of programming. Fundamental data structures.	
5	**Lecture 5. Introduction to C.** Introduction to the C programming language. Alphabet and language dictionary. Identifiers. Basic data types. Type modifiers. Lit.: [**1, C.7-18; 2, C. 6-12; 5-14**]	2
6	**Lecture 6. The structure of the C programming language.** Constants. Prefix and suffix forms. Expressions and operations. Decimal, octal and hexadecimal representation. Identifiers. Keywords. Comments. Language standards C. Lit.: [**1, C.7-18; 2, C. 13-20; 5-14**]	2
7	**Lecture 7. The structure of the C programming language. Standard functions** The structure of the C language program. Prototypes of functions. Guidelines for including the contents of files. Basic input-output tools. Function output specifications. Data entry using standard functions.	2

	Lit.: [**1, C.19-30; 2, C. 27-48; 5-14**] **Topic 4.** Structural programming. Constructions of programming languages.	
8	**Lecture 8. General information about operators** General information about operators. Folded operator. Empty operator. Expressions. Operation sign. Types of arithmetic operations. Priority of operations. Lit.: [**1, C.31-111; 2, C. 17-26; 5-14**]	2
9	**Lecture 9. General information about operators** Branching operator, its forms. Logical operations. Compiled assignment operators. Conditional selection operation. Lit.: [**1, C.85-111; 2, C. 17-27; 5-14**]	2
10	**Lecture 10. General information about operators** Using a comma. Cycle operators. Operator with parameter and its features. Operators of cycle-while and cycle-to, their differences. Lit.: [**1, C.85-111; 2, C. 17-27; 4-14**]	2
11	**Lecture 11. Functions in the C language** Announcement and definition of functions. Arguments, parameters, examples. Transfer parameters by value and by link. Return from function and return values. Arithmetic functions. Recursion. Recursive calls. Direct and indirect recursion. Basic principles of structural programming. Different types of functions. Lit.: [**1, C.183-234; 2, C. 27-47; 4-14**]	2
12	**Lecture 12. Arrays** Description of arrays. Arrays of elements. Array declarations. Access to array components, one- and n-dimensional arrays. Examples. Sorting and search methods. Computational complexity of algorithms. Lit.: [**1, C.125-140; 2, C. 29; 3, C.44-59, 105-117; 4-14**]	2
13	**Lecture 13. Strings** Strings. Concepts and announcements. Indexes of string elements. Merge strings. Comparison of lines. Input - output lines. Other functions for string processing. Examples. Lit.: [**1, C.141-163; 2, C. 52-53; 4-14**]	2
14	**Lecture 14. Files** Files. Definition. File types. General rules for all file types. Text files. Algorithm for creating a text file, reading, writing, writing to a text file. Features of application of standard functions and procedures when working with files. Lit.: [**1, C.141-163; 2, C. 52-53; 4-14**] **Topic 5.** Programming dynamic data structures. Algorithms and data structures.	2
15	**Lecture 15. Pointers**	2

	The concept of pointers. Creating pointers. Announcement of pointers. Pointers and types of variables. Pointers and arrays. Transfer arrays to functions. Dynamic data structures. Lit.: [**1, C.112-124; 2, C. 34-35; 4-14**]	
16	**Lecture 16. Structures, associations, and custom data types** The simplest structures. Complex structures. Arrays of structures. Lit.: [**1, C.112-124; 2, C. 30-31; 4-14**]	2
17	**Lecture 17. Additional information about pointers** Pointers to pointers. The difference between dynamic data and static. Stack, queue, binary tree. Lit.: [**1, C.246-295; 2, C. 35; 4-14**]	2
	Total for the 1th semester:	34

5.2 Content of laboratory works

Number	Topic of the laboratory class	Number of hours
	First semester	
1	Linear algorithms. Implementation in the C language.	8
2	Branched algorithms. Implementation in the C language.	8
3	Programming of the loop algorithms. Implementation in the C language.	8
4	Arrays. Working with one-dimensional arrays. Implementation in the C language.	8
5	Two-dimensional arrays. Implementation in the C language.	8
6	Functions. Implementation in the C language.	8
7	Strings. Implementation in the C language.	8
8	Streams and files. Working with text files. Their implementation in the C language.	8
9	Final lesson.	4
	Total for the 1th semester	68

5.3 Content of practical classes

Number	Topic of the practical class	Number of hours
	First semester	
1	**Practical classes №1-2.** Linear algorithms.	2
2	**Practical classes №3-4.** Branched algorithms.	2
3	**Practical classes №5-6.** Programming of the loop algorithms.	2
4	**Practical classes №7-8.** Arrays. Working with one-dimensional arrays.	2
5	**Practical classes №9-10.** Two-dimensional arrays.	2
6	**Practical classes №11-12.** Functions.	2
7	**Practical classes №13-14.** Strings.	2
8	**Practical classes №15-16.** Streams and files. Working with text files. Structures.	2
9	Final class	1
	Total for the 1^{th} semester	17

5.4 Content of independent (individual) work

The volume of independent work in the discipline "Programming" is 164 hours. It includes the study of lecture material, preparation for laboratory work and their defense, preparation for practical work, preparation for current control.

Week number	Type of independent work	Number of hours
	First semester	
1	Elaboration of lecture material. Preparation for laboratory work №1 and practical work №1. Independent work on program development for laboratory work №1.	21
2	Elaboration of lecture material. Preparation for laboratory work №2 and practical work №2. Independent work on program development for laboratory work №2.	21
3	Elaboration of lecture material. Preparation for laboratory work №3 and practical work №3. Independent work on program development for laboratory work №3.	21
4	Elaboration of lecture material. Preparation for laboratory work №4 and practical work №4. Independent work on program development for laboratory work №4.	21
5	Elaboration of lecture material. Preparation for laboratory work №5 and practical work №5. Independent work on program development for laboratory work №5.	21
6	Elaboration of lecture material. Preparation for laboratory work №6 and practical work №6. Independent work on program development for laboratory work №6.	21
7	Elaboration of lecture material. Preparation for laboratory work №7 and practical work №7. Independent work on program development for laboratory work №7.	21
8	Elaboration of lecture material. Preparation for laboratory work №8 and practical work №8. Independent work on program development for laboratory work №8.	14
	Total for the 1th semester:	161

6. TEACHING METHODS

The educational process for the discipline is based on the use of both traditional and modern methods. Specifically, the methods used include: lectures (employing problem-based learning and visualization techniques); laboratory and practical classes (utilizing information technology methods, modern integrated programming environments, workshops, and practical exercises), and independent work (home practical assignments), aimed at acquainting students with contemporary methods of algorithm and software development and their practical application in software design and development.

7. ASSESSMENT FORMS AND METHODS

Current control is carried out during lectures, practical and laboratory classes, as well as on the days of control activities established by the work plan of the discipline. Semester tests are conducted in the form of exam, credit. At the same time, the results of the current control are taken into account when deriving the final assessment.

The teaching of the discipline uses such types of classes as lectures, laboratory work, practical work, course design, individual counseling and guidance of independent student work.

Each type of work required for the discipline is graded on a four-point scale. The semester final grade is defined as the weighted average of all types of academic work performed and passed positively, taking into account the weighting factor. Weights vary depending on the structure of the discipline and the importance of its individual types of work. A student who scored a positive weighted average score for the current work and did not pass the final test (exam) is considered to have failed.

When assessing students' knowledge, various means of control are used, in particular: oral examination before admission to laboratory work - is carried out at the beginning; mastering the theoretical material on the topic is checked by test control; the quality of performance, acquisition of theoretical knowledge and practical skills is checked by defending each laboratory work in accordance with the work program of the discipline and the working curriculum.

The assessment, which is set for the laboratory lesson, consists of the following elements: oral examination of students before admission to laboratory work; knowledge of theoretical material on the topic; the quality of the protocol and graphic part; the student's ability to justify constructive decisions; timely protection of laboratory work. To complete the discipline program, the student must receive 8 grades for laboratory work.

The term of defense of laboratory work is considered timely if the student defended it in the next lesson after the work.

The student must complete the missed laboratory lesson in the laboratories of the department within the period set by the teacher with registration in the relevant journal of the department, but not later than two weeks before the end of theoretical classes in the semester.

When assessing students' knowledge, the teacher is guided by the following criteria.

The student receives an "excellent" grade on the ECTS-A scale (see grading scale) for deep and complete mastery of the content of educational material in which he is easily oriented, the conceptual apparatus, the ability to connect theory with practice, solve practical problems, express and substantiate their judgments. Excellent assessment involves a competent, logical presentation of the answer (both orally and in writing), quality exterior design. The student must acquire practical skills in compiling various algorithms and developing programs based on these algorithms. The grade "excellent" is given to a student who has deeply mastered the operators, functions and procedures of the C language and is able to rationally apply them, knows the techniques and is able to use them in compiling algorithms and programs. The student should not hesitate to change the question, should make detailed and generalizing conclusions.

The student receives a grade of "good" on the ECTS-B scale for full mastering of the study material, mastery of the conceptual apparatus, orientation in the studied material, conscious use of knowledge to solve practical problems, competent presentation of the answer, but in the content and

form of the answer (errors), vague wording of patterns, etc. The student's answer should be based on independent thinking.

The student receives a grade of "good" on the ECTS - C scale for the correct answer with one significant error.

Grades "satisfactory", according to the ECTS - D scale, deserves a student who has shown knowledge of the basic curriculum in the amount necessary for further study and practical activities in the profession that copes with the practical tasks provided by the program. As a rule, the student's answer is based on the level of reproductive thinking, the student knows little about the structure of the course, makes mistakes in the answer, mastered and acquired practical skills in compiling programs, but made inaccuracies. He hesitates in answering the modified question, however, the student has the knowledge that allows him under the guidance of the teacher to eliminate inaccuracies in the answer.

Grades "satisfactory", according to the ECTS - E scale, the student deserves for incomplete mastery of the program material, but he gained knowledge and acquired practical skills in developing programs in the C language.

The grade "unsatisfactory", according to the ECTS - FX scale, is given when the student has disparate, unsystematic knowledge, can not distinguish between primary and secondary, errors in defining concepts, distorts their meaning, chaotically and uncertainly presents material, can not use knowledge in solving practical tasks. As a rule, a grade of "unsatisfactory" is given to a student who cannot continue his studies without additional knowledge of the course.

The grade "unsatisfactory", according to the ECTS - F scale, is given to the student for complete ignorance and misunderstanding of the study material or refusal to answer and involves re-learning the student in the discipline.

Each type of work is evaluated on a four-point scale. The semester final grade is defined as the weighted average of all types of work.

Structuring the course by types of work and assessing learning outcomes for full-time students in the semester according to weighing coefficients

Auditory work	**Independent work**			**Semester control (exam)**
Laboratory work	Test control		Control work	Final control measure
No. 1 - 12	TC1 (T1-3)	TC2 (T4-6)	1	1
0.3	**0. 2**		**0.1**	**0.4**

Assessing tests. The thematic test for each student consists of twenty-five test items, each of which is graded one point. The maximum amount of points that a student can score is 25.

The assessment is based on a four-point scale.

The correspondence of the scored points for the test to the grade given to the student is presented in the table below.

Scored points for the test	1–11	12–14	15–22	23-25
Grade	2	3	4	5

30 minutes are allotted for testing. Testing is performed using a modular learning environment MOODLE. The student registers the correct answers online in the modular environment MOODLE. After 30 minutes, students complete the test and send their answers to the server. The teacher announces the results of testing according to the journal of assessments of the

modular environment MOODLE.

If a student receives a negative grade, he must retake it in the prescribed manner, but always before the next control.

The final semester grade according to the national scale and the ECTS scale is set in an automated mode after entering all the grades in the electronic journal. The ratios of the domestic assessment scale and the ECTS assessment scale are given in the following table.

To move from the national grade to the ECTS scale, you need to find the arithmetic mean of the national scale, multiply it by the appropriate weighting factor and, adding all the components, get the sum of points that will determine a particular ECTS grade.

Correspondence of the national and ECTS grading scales

ECTS grade	***Institutional score scale***	***Institutional grade***		***Assessment criteria***
A	4,75-5,00	5	Passed	**Excellent** – deep and complete mastery of educational material and demonstrating relevant skills and abilities.
B	4,25-4,74	4		**Good** – complete knowledge of the material with a few minor errors.
C	3,75-4,24	4		**Good** – correct answer in general with two to three significant errors.
D	3,25-3,74	3		**Satisfactory** – incomplete mastery of the program material but sufficient for practical activities in the professional field.
E	3.00-3,24	3		**Satisfactory** – incomplete mastery of the program material that meets the minimum assessment criteria.
FX	2,00-2,99	2	Failed	**Unsatisfactory** – unsystematic knowledge and inability to continue studies without additional knowledge of the course.
F	0,00-1,99	2		**Unsatisfactory** – serious further work is needed and the course is to be retaken.

8. QUESTIONS FOR STUDENTS' SELF-CONTROL

1. Information technologies and systems
2. History of computer technology
3. History of operating systems, their main common implementations
4. Information and its presentation
5. Binary arithmetic
6. General structure of the computer
7. Computer systems and their components
8. The concept of algorithms and their properties
9. Step-by-step detailing method
10. Recording of the algorithm in the form of block diagrams
11. Forms of representation of algorithms
12. Types of basic structures of algorithms
13. Alphabet and dictionary of language C
14. Basic data types
15. Constants
16. Identifiers
17. Keywords
18. Comments
19. Language standards C
20. The structure of the language program C
21. #include Directive
22. The main means of input-output
23. General information about operators
24. Expressions
25. Branching operator
26. Logical operations
27. Compound assignment operators
28. Conditional selection operation
29. Using a comma
30. Cycle operators
31. Cycle with postcondition, example program
32. FOR cycle
33. Cycle with a prerequisite, an example of a program
34. Announcement and definition of functions
35. Recursive call
36. Arrays of elements
37. Multidimensional arrays
38. The method of pairwise permutation of elements
39. The method of the smallest elements
40. Binary search method
41. The concept of pointers
42. Creating pointers
43. Announcement of pointers
44. Pointers and types of variables
45. Pointers and arrays
46. Transfer of arrays in functions
47. The concept of strings
48. Ways to allocate memory
49. Functions of input and output of characters and lines
50. Increment and decrement operations
51. Operator continue
52. Break operator
53. Return the values of the function using the return operator

54. Prototypes
55. Additional assignment operations
56. Object code files, executables, and libraries
57. IF branching operator
58. Compound literals.
59. Interaction with files.
60. Standard files.
61. The fopen () function.
62. File compression program.
63. Input - output files.
64. Random access to files
65. The simplest data structures
66. Complex structures.
67. Arrays of structures.
68. Initialization of structures.

9. TEACHING AND LEARNING MATERIALS

The educational process in the discipline of "Programming" is fully and sufficiently provided with the necessary educational and methodological literature.

10. RECOMMENDED LITERATURE

1. Seacord R. Effective C: An Introduction to Professional C Programming. - San Francisco, California, No Starch Press, 2020. - 272 p.
2. Gustedt J. Modern C. - Shelter Island, Manning, 2020. – 408 p.
3. King K.N. C programming: a modern approach. - W. W. Norton & Company, 2020. – 832 p.
4. Harwani B.M. Practical C Programming: Solutions for modern C developers to create efficient and well-structured programs. – Birmingham, Packt Publishing, 2020. – 616 p.
5. Hubert H.W. Intermediate C programming for the PIC microcontroller: simplifying embedded programming. - Berkeley, CA, Apress L. P., 2020. – 318 p.
6. Winkle L.Van. Hands-On Network Programming with C: Learn socket programming in C and write secure and optimized network code. – Birmingham, Packt Publishing, 2019. – 478 p.
7. Harwani B.M. C programming cookbook: over 40 recipes exploring data structures, pointers, interprocess communication, and database in C. - Birmingham, UK, Packt Publishing, 2019. – 344 p.
8. Joyce P. Numerical C: applied computational programming with case studies. - Berkeley, CA, Apress, 2019. – 312 p.
9. McGrath M. C Programming in easy steps: Updated for the GNU Compiler version 6.3.0 and Windows 10 5th Edition. - In Easy Steps Limited, 2018. – 192 p.
10. Kane W, C programming for business. Book one, Introduction to problem solving in C language. - Choice Publishing, 2017. – 284 p.

11 . INFORMATION RESOURCES

1. MOODLE Learning Platform. Access to the resource https://msn.khmnu.edu.ua.
2. University Electronic Library. Access to the resource: http://library.khmnu.edu.ua.
3. University Repository. Access to the resource https://elar.khmnu.edu.ua/home.